#4

OIPE JC48
OCT 1 6 2000
PATENT & TRADEMARK OFFICE

EXHIBIT

Various materials showing that "address translation circuitry" is a term well-known in the art

MOTOROLA

MC88110

Second Generation RISC Microprocessor User's Manual

8.1.5 Address Translation Modes

The MMUs of the MC88110 provide flexibility in controlling the mechanism for address translation. There are four address translation modes available to an operating system: the identity translation mode, the block-exclusive translation mode, the page-exclusive translation mode, and the combined BATC/PATC translation mode. Each mode is selected independently for the IMMU and DMMU. See **8.2 Selection of Address Translation Mode** for information on how the translation mode is selected.

In the identity translation mode, physical addresses are identical to logical addresses, so the MMU passes logical addresses directly to the cache or BIU with no address translation. Access control for accesses performed in this mode is regulated by the appropriate bits in the MMU control registers (see **8.9 MMU/Cache Control Registers**). Identity translation is always selected if the DEBUG signal is asserted, overriding software selection of other address translation modes.

In the block-exclusive translation mode, the MMU uses only the BATC to translate logical addresses into physical addresses and to obtain access control bits. This mode is often useful while executing programs that are permanently resident or completely swapped into physical memory prior to being executed. If an appropriate entry is not found in the BATC (BATC miss), then identity translation occurs, and no fault or exception is taken.

In the page-exclusive translation mode, the MMU translates logical addresses into physical addresses and obtains access control bits using only the PATC. In the event of a PATC miss, the MMU performs a hardware table search operation or causes the appropriate ATC miss exception to be taken (see **8.4.4.1 Software Table Search Operations**), depending on whether hardware table searching is enabled. If the hardware table search operation succeeds, the MMU automatically loads the necessary page descriptor into the PATC and uses it to complete the address translation; if the hardware table search operation fails, the MMU causes the instruction or data memory access exception (see **8.7 MMU/Cache Faults**) to be taken.

In combined block/page translation mode, the MMU attempts to translate logical addresses and obtain access control bits simultaneously in the BATC and in the PATC. This mode is often useful if a program executes in a demand paged virtual memory environment, but needs to access some permanently resident instructions or data or perform memory mapped I/O.

For example, an application with paged code/data may still need to access a frame buffer or call functions in a large shared library. The register file within a memory mapped I/O device, a frame buffer, or a shared library is permanently resident at a known physical address and can be described permanently with a single block descriptor, even if the remainder of the program's code and data are dynamically allocated in physical memory. In the combined block/page translation mode, in the event of a BATC hit, the MMU operates as in block-exclusive translation mode. In the event of a BATC miss, the MMU operates as in page-exclusive translation mode. Therefore, if a BATC hit and a PATC hit both occur for a logical address, the BATC entry is used.

intel®

Intel Architecture Software Developer's Manual

Volume 3: System Programming Guide

NOTE: The *Intel Architecture Developer's Manual* consists of three books: *Basic Architecture*, Order Number 243190; *Instruction Set Reference Manual*, Order Number 243191; and the *System Programming Guide*, Order Number 243192.
Please refer to all three volumes when evaluating your design needs.

1997

3.6.2. Page Tables and Directories

The information that the processor uses to translate linear addresses into physical addresses (when paging is enabled) is contained in four data structures:

- Page directory—An array of 32-bit page-directory entries (PDEs) contained in a 4-KByte page. Up to 1024 page-directory entries can be held in a page directory.
- Page table—An array of 32-bit page-table entries (PTEs) contained in a 4-KByte page. Up to 1024 page-table entries can be held in a page table. (Page tables are not used for 2-MByte or 4-MByte pages. These page sizes are mapped directly from one or more page-directory entries.)
- Page—A 4-KByte, 2-MByte, or 4-MByte flat address space.
- Page-Directory-Pointer Table—An array of four 64-bit entries, each of which points to a page directory. This data structure is only used when the physical address extension is enabled (see Section 3.8., "Physical Address Extension").

These tables provide access to either 4-KByte or 4-MByte pages when normal 32-bit physical addressing is being used and to either 4-KByte or 2-MByte pages when extended (36-bit) physical addressing is being used. Table 3-3 shows the page size and physical address size obtained from various settings of the paging control flags. Each page-directory entry contains a PS (page size) flag that specifies whether the entry points to a page table whose entries in turn point to 4-KByte pages (PS set to 0) or whether the page-directory entry points directly to a 4-MByte or 2-MByte page (PSE or PAE set to 1 and PS set to 1).

Table 3-3. Page Sizes and Physical Address Sizes

PG Flag, CR0	PAE Flag, CR4	PSE Flag, CR4	PS Flag, PDE	Page Size	Physical Address Size
0	X	X	X	—	Paging Disabled
1	0	0	X	4 KBytes	32 Bits
1	0	1	0	4 KBytes	32 Bits
1	0	1	1	4 MBytes	32 Bits
1	1	X	0	4 KBytes	36 Bits
1	1	X	1	2 MBytes	36 Bits

3.6.2.1. LINEAR ADDRESS TRANSLATION (4-KBYTE PAGES)

Figure 3-12 shows the page directory and page-table hierarchy when mapping linear addresses to 4-KByte pages. The entries in the page directory point to page tables, and the entries in a page table point to pages in physical memory. This paging method can be used to address up to 2^{20} pages, which spans a linear address space of 2^{32} bytes (4 GBytes).

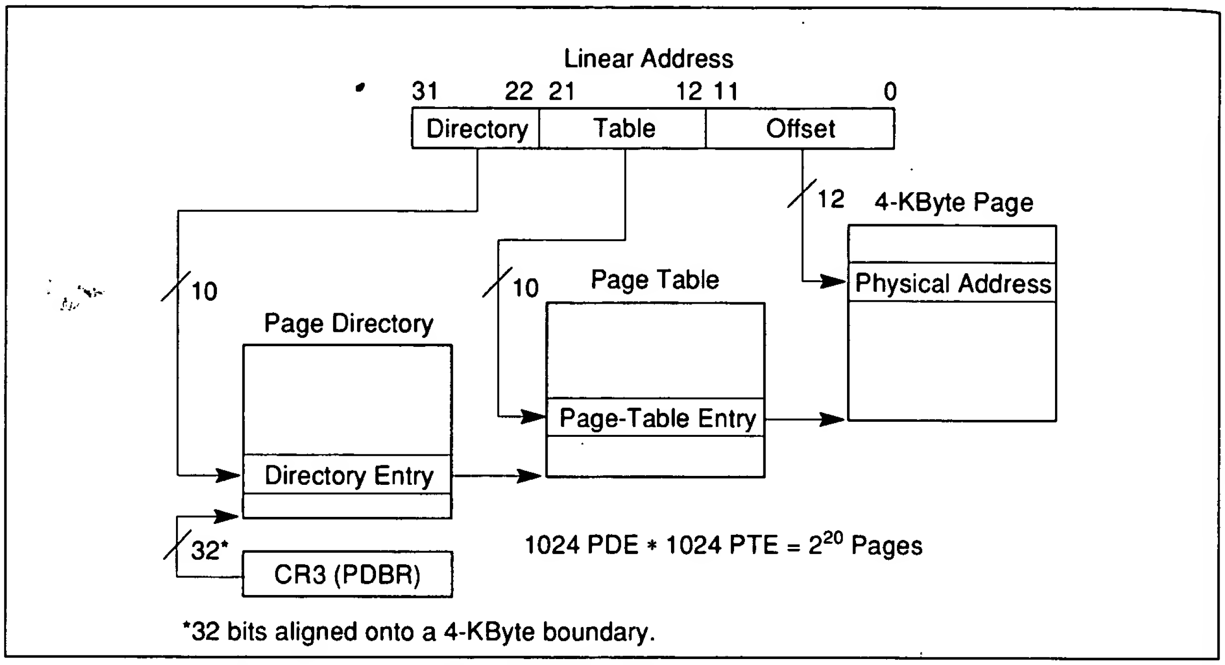

Figure 3-12. Linear Address Translation (4-KByte Pages)

To select the various table entries, the linear address is divided into three sections:

- Page-directory entry—Bits 22 through 31 provide an offset to an entry in the page directory. The selected entry provides the base physical address of a page table.
- Page-table entry—Bits 12 through 21 of the linear address provide an offset to an entry in the selected page table. This entry provides the base physical address of a page in physical memory.
- Page offset—Bits 0 through 11 provides an offset to a physical address in the page.

Memory management software has the option of using one page directory for all programs and tasks, one page directory for each task, or some combination of the two.

3.6.2.2. LINEAR ADDRESS TRANSLATION (4-MBYTE PAGES)

Figure 3-12 shows how a page directory can be used to map linear addresses to 4-MByte pages. The entries in the page directory point to 4-MByte pages in physical memory. This paging method can be used to map up to 1024 pages into a 4-GByte linear address space.

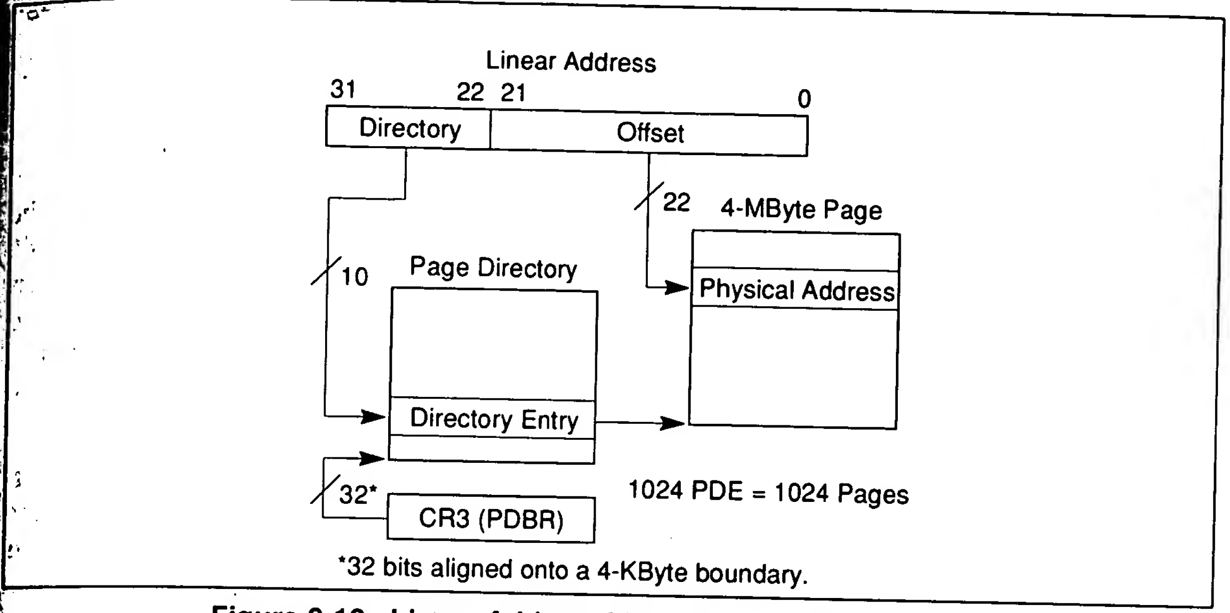

Figure 3-13. Linear Address Translation (4-MByte Pages)

The 4-MByte page size is selected by setting the PSE flag in control register CR4 and setting the page size (PS) flag in a page-directory entry (see Figure 3-14). With these flags set, the linear address is divided into two sections:

- Page directory entry—Bits 22 through 31 provide an offset to an entry in the page directory. The selected entry provides the base physical address of a 4-MByte page.
- Page offset—Bits 0 through 21 provides an offset to a physical address in the page.

NOTE

(For the Pentium® processor only.) When enabling or disabling large page sizes, the TLBs must be invalidated (flushed) after the PSE flag in control register CR4 has been set or cleared. Otherwise, incorrect page translation might occur due to the processor using outdated page translation information stored in the TLBs. See Section 9.9., "Invalidating the Translation Lookaside Buffers (TLBs)", for information on how to invalidate the TLBs.

3.6.2.3. MIXING 4-KBYTE AND 4-MBYTE PAGES

When the PSE flag in CR4 is set, both 4-MByte pages and page tables for 4-KByte pages can be accessed from the same page directory. If the PSE flag is clear, only page tables for 4-KByte pages can be accessed (regardless of the setting of the PS flag in a page-directory entry).

A typical example of mixing 4-KByte and 4-MByte pages is to place the operating system or executive's kernel in a large page to reduce TLB misses and thus improve overall system performance. The processor maintains 4-MByte page entries and 4-KByte page entries in separate

Alpha AXP Architecture Reference Manual
Second Edition

Richard L. Sites
Richard T. Witek

Contributing Authors for the Second Edition

Wayne M. Cardoza
Anton Chernoff
John A. DeRosa
Daniel W. Dobberpuhl
John H. Edmondson
Kent Glossop
Henry N. Grieb
Richard B. Grove
Michael S. Harvey
Steven O. Hobbs
Robert F. Hoffman
Nancy P. Kronenberg
Raymond J. Lanza
Derrick R. Meyer
Stephen J. Morris
Joseph Notarangelo
William B. Noyce
Mary H. Payne
Audrey R. Reith
Robert M. Supnik
Benjamin J. Thomas

Digital Press
Boston • Oxford • Melbourne • Singapore • Toronto • Munich • New Delhi • Tokyo

each processor mode. The code allows a choice of read or write protection for each processor mode.

- Each mode's access can be read/write, read-only, or no-access.
- Read and write accessibility are specified independently.
- The protection of each mode can be specified independently.

The protection code is specified by 8 bits in the PTE (see Table 3–2).

The OpenVMS AXP architecture allows a page to be designated as execute only by setting the read enable bit for the access mode and by setting the fault on read and write bits in the PTE.

3.6.3 Access Violation Fault

An Access Violation fault occurs if an illegal access is attempted, as determined by the current processor mode and the page's protection field.

3.7 Address Translation

The page tables can be accessed from physical memory, or (to reduce overhead) through a mapping to a linear region of the virtual address space. All implementations must support the virtual access method and are expected to use it as the primary access method to enhance performance.

The following sections describe both access methods.

3.7.1 Physical Access for Page Table Entries

Physical address translation is performed by accessing entries in a three-level page table structure. The Page Table Base Register (PTBR) contains the physical Page Frame Number of the highest level (Level 1) page table. Bits <level1> of the virtual address are used to index into the first level page table to obtain the physical page frame number of the base of the second level (Level 2) page table. Bits <level2> of the virtual address are used to index into the second level page table to obtain the physical page frame number of the base of the third level (Level 3) page table. Bits <level3> of the virtual address are used to index the third level page table to obtain the physical Page Frame Number (PFN) of the page being referenced. The PFN is concatenated with virtual address bits <byte_within_page> to obtain the physical address of the location being accessed.

If part of any page table resides in I/O space, or in nonexistent memory, the operation of the processor is UNDEFINED.

If the first-level or second-level PTE is valid, the protection bits are ignored; the protection code in the third-level PTE is used to determine accessibility. If a first-level or second-level PTE is invalid, an Access Violation occurs if the PTE<KRE> equals zero. An Access Violation on a first-level or second-level PTE implies that all lower-level page tables mapped by that PTE do not exist.

PA-RISC 2.0 Architecture

Gerry Kane

Prentice Hall PTR
Upper Saddle River, New Jersey 07458

absolute address is formed by taking the offset and forcing the most significant 2 bits to 0.

This way of forming absolute address for memory allows software more flexibility in address space layout. Note that this has no impact on machine implementing fewer than 62 bits of physical address.

Address Resolution and the TLB

Virtual addresses are translated to absolute addresses using a hardware structure called the **Translation Lookaside Buffer** (TLB). A TLB accepts a Virtual Page Number and returns the corresponding Physical Page Number. The TLB is organized as two parts. The instruction TLB (ITLB) is only used for instruction references, while the data TLB (DTLB) is only used for data references. A system may implement a **combined** TLB which is used for both instruction and data references.

A TLB is typically not large enough to hold all the current translations. Translations for all pages in memory are stored in a memory structure called the **Page Table**. Multiple page sizes are supported, from 4 Kbytes to 64 Mbytes. This allows large contiguous regions to be mapped with a single TLB entry. This increases the virtual address range of the TLB, thereby minimizing the virtual address translation overhead.

Given a virtual address, the selected TLB is searched for an entry matching the Virtual Page Number. If the entry exists, the 38 to 52-bit Physical Page Number (contained in the TLB entry) is concatenated with the original 12 to 26-bit page offset (depending on the page size in the matching entry) to form a 64-bit absolute address. If no such entry exists, the TLB is updated by either software TLB miss handling or hardware TLB miss handling.

In systems with software TLB miss handling, a TLB miss fault interruption routine performs the translation, explicitly inserts the translation and protection fields into the appropriate TLB, and restarts the interrupted instruction. To insure the completion of instructions, the TLBs must be organized to simultaneously hold all necessary translations.

In implementations that provide hardware for TLB miss handling, the hardware attempts to find the virtual to physical page translation in the Page Table. If the hardware is successful, it inserts the translation and protection fields into the appropriate instruction or data TLB. No interruption occurs in this case. If hardware is not successful, due to a search of the Page Table that was not exhaustive or due to the appropriate translation not existing in the Page Table, an interruption occurs so that the software can complete the process.

The translation lookaside buffer performs other functions in addition to the basic address translation. The other functions include access control, program debugging support and operating system support for virtual memory. Figure 3-9 summarizes the information maintained for each TLB entry.

SPARC RISC USER'S GUIDE

ROSS Technology, Inc.
A Cypress Semiconductor Company

ROSS Technology, Inc.	• 7748 Hwy. 290 West	• Austin	• TX 78736	• 512-448-8968
Cypress Semiconductor Corporation	• 3901 North First Street	• San Jose	• CA 95134	• 408-943-2600

4.1.1 Translation Lookaside Buffer (TLB)

The CY7C604/605 uses a 64-entry fully associative TLB for address translation. The TLB consists of two sections: a virtual section and a physical section, as shown in *Figure 4–2*. The virtual section is compared against the virtual address and the contents of the context register. A content addressable memory (CAM) is used as the virtual section of the TLB. The CAM provides simultaneous comparison of all 64 TLB entries with the current virtual address and context. The physical section of the TLB is a RAM array, and its entries are addressed by a valid compare output from a CAM entry. If a CAM entry matches the virtual address and context, the corresponding RAM entry in the TLB provides the physical address for use by the CY7C604/605.

The virtual section of a TLB entry consists of 20 bits of virtual address (VA(31:12)) and a 12-bit context number (CXN(11:0)). The physical section of a TLB entry consists of a 24-bit physical page number (PPN(35:12)), a cacheable bit (C), a modified bit (M), a three-bit field for page access-level protection (ACC(2:0)), a two-bit short translation field (ST(1:0)), and one valid bit (V).

As described by the SPARC reference MMU specification, bits 31 through 12 of the virtual address are translated to an expanded physical address using bits 35 through 12. The translation of these bits depends upon the ST field of the TLB entry (or PTE) and the MMU operation mode (refer to page 4–13). Bits 11 through 0 of the virtual address are not translated, and are defined as the page offset for the 4-kbyte memory page.

A TLB entry (PTE) can be defined to map a virtual address into one of four sizes of addressing regions using the ST field. The four sizes of addressing regions are: 4-kbyte, 256-kbyte, 16-Mbyte, or 4-Gbyte. *Table 4–1* illustrates the values assigned to the ST(1:0) field.

The value of the short translation bits affects both the addresses generated using the TLB entry and the virtual addresses allowed to match with the TLB entry. The virtual address supplied by the integer unit is divided into four fields: index 1, index 2, index 3, and page offset, as illustrated in *Figure 4–3*. For ST = (1,1) (4-Gbyte addressing range), only the context register is used to match a TLB entry. Setting ST = (1,1) essentially causes the CAM array to ignore the index 1, 2, and 3 fields of the virtual address. Consequently, the address generated using the TLB entry only supplies the upper four bits of the 36-bit physical address. Index 1, 2, and 3 fields, along with the page offset, are passed along to the physical address unchanged.

The three remaining values of the ST field "turn on" comparison of the three index fields. The index fields that are required to match a TLB entry also become the fields that are replaced by the TLB entry during virtual to physical translation. Setting ST = (1,0), (16-Mbyte addressing region), requires the TLB to match the context and index 1 fields of the virtual address to the TLB entry. The TLB entry with ST = (1,0) will supply the upper four address bits and replace the index 1 field of the virtual address with a physical address field. The index 2, 3, and page offset fields are passed along to the physical address from the virtual address. Setting ST = (0,1) and (0,0) adds index 2 and index 3 fields to the comparison, respectively. Setting ST = (0,0) causes the TLB to require matching of the context, index 1, 2, and 3, and will replace all but the page offset when translating the virtual address.

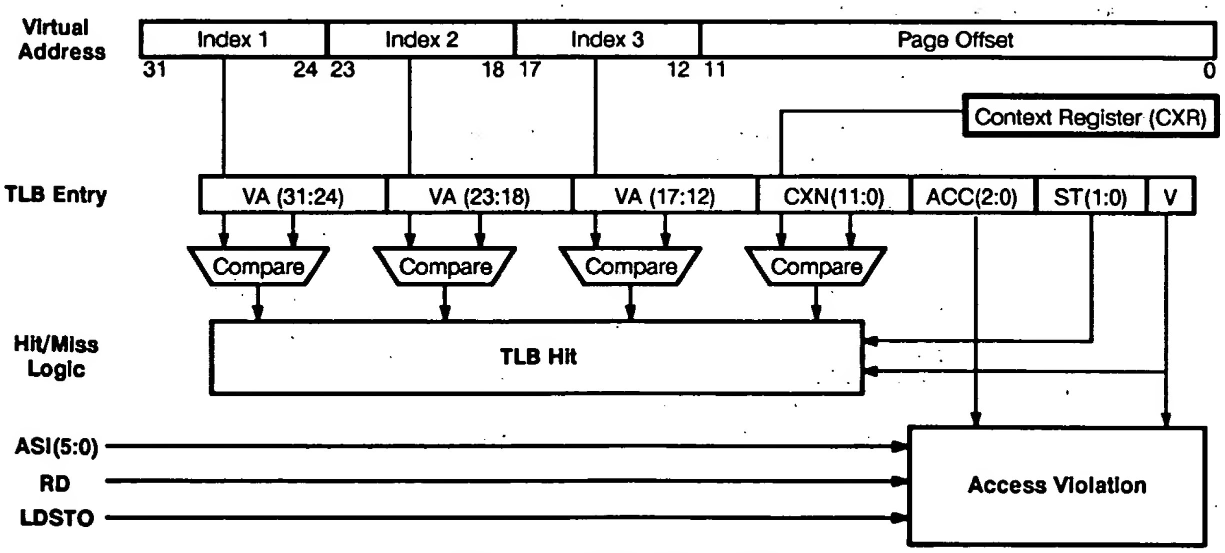

Figure 4–3. Address Comparison

VAX Architecture Reference Manual

Edited by
Timothy E. Leonard

Contributing Authors

Dileep P. Bhandarkar
Peter F. Conklin
David N. Cutler
Thomas W. Eggers
Thomas N. Hastings
Richard I. Hustvedt
Judson S. Leonard
Peter Lipman
Thomas Rarich
David P. Rodgers
Stephen Rothman
William D. Strecker
Theodore B. Taylor

digital

DECbooks

```
PA = VA<29:0> modulo (2**number of PA bits)
```

There is no page protection: all accesses are allowed in all modes. No modify bit is maintained.

Note, however, that references to nonexistent memory may cause unexpected results when memory management is disabled. The accessibility of nonexistent memory is UNPREDICTABLE when memory management is disabled (see the PROBE instructions). In addition, a processor may have an instruction buffer that prefetches instructions before execution. If the instruction stream comes within 512 bytes of nonexistent memory when memory management is disabled, prefetcher references may cause UNDEFINED behavior.

ADDRESS TRANSLATION

When MME is a 1, address translation and access control are on. The processor uses the following to determine whether an intended access is allowed:

1. The virtual address, which is used to index a page table
2. The intended access type (read or write)
3. The current privilege level from the processor status longword, or kernel level for page table mapping references.

If the access is allowed and the address can be mapped (the page table entry is valid), the result is the physical address corresponding to the specified virtual address.

The intended access is READ if the operation to be performed is a read. The intended access is WRITE if the operation to be performed is a write. If the operation to be performed is a modify (that is, read followed by write), the intended access is specified as a WRITE.

If an operand is an address operand, then no reference is made. Hence the page need not be accessible and need not even exist.

Page Table Entry

The CPU uses a page table entry (PTE), shown in Figure 4.4, to translate virtual addresses to physical addresses. The fields of the PTE are described in Table 4.1.

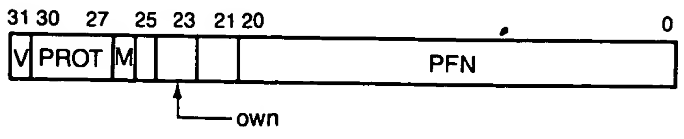

Figure 4.4
Page Table Entry

Computer Architecture A Quantitative Approach

David A. Patterson
UNIVERSITY OF CALIFORNIA AT BERKELEY

John L. Hennessy
STANFORD UNIVERSITY

With a Contribution by
David Goldberg
Xerox Palo Alto Research Center

MORGAN KAUFMANN PUBLISHERS, INC.
SAN MATEO, CALIFORNIA

belonging just to that process. Most forms of virtual memory also reduce the time to start a program, since not all code and data need be in physical memory before a program can begin.

While virtual memory is essential for current computers, sharing is not the reason virtual memory was invented. In former days if a program became too large for physical memory, it was up to the programmer to make it fit. Programmers divided programs into pieces and then identified the pieces that were mutually exclusive. These *overlays* were loaded or unloaded under user program control during execution, with the programmer ensuring that the program never tried to access more physical main memory in the machine. As one can well imagine, this responsibility eroded programmer productivity. Virtual memory, invented to relieve programmers of this burden, automatically managed the two levels of the memory hierarchy represented by main memory and secondary storage.

In addition to sharing protected memory space and automatically managing the memory hierarchy, virtual memory also simplifies loading the program for execution. Called *relocation*, this procedure allows the same program to run in any location in physical memory. (Prior to the popularity of virtual memory, machines would include a relocation register just for that purpose.) An alternative to a hardware solution would be software that changed all addresses in a program each time it was run.

Several general memory-hierarchy terms from Section 8.3 apply to virtual memory, while some other terms are different. *Page* or *segment* is used for block, and *page fault*, or *address fault*, is used for miss. With virtual memory, the CPU produces *virtual addresses* that are translated by a combination of hardware and software to *physical addresses*, which can be used to access main memory. This process is called *memory mapping* or *address translation*. Today, the two memory hierarchy levels controlled by virtual memory are DRAMs and magnetic disks. Figure 8.21 shows a typical range of memory hierarchy parameters for virtual memory.

Block (page) size	512 – 8192 bytes
Hit time	1–10 clock cycles
Miss penalty	100,000 – 600,000 clock cycles
(Access time)	(100,000–500,000 clock cycles)
(Transfer time)	(10,000–100,000 clock cycles)
Miss rate	0.00001%–0.001%
Main memory size	4 MB – 2048 MB

FIGURE 8.21 Typical ranges of parameters for virtual memory. These figures, contrasted with the values for caches in Figure 8.5 (page 408), represent increases of 10 to 100,000 times.